A Branch and Cut Algorithm for Nonconvex Quadratically Constrained Quadratic Programming

Charles Audet, Pierre Hansen, Brigitte Jaumard, and Gilles Savard

CRPC-TR99783-S
January 1999

Center for Research on Parallel Computation
Rice University
6100 South Main Street
CRPC - MS 41
Houston, TX 77005

Submitted January 1999

A Branch and Cut Algorithm for Nonconvex Quadratically Constrained Quadratic Programming

Charles Audet
Rice University, CAAM -MS134
6100 Main Street
Houston, Texas, 77005-1892 USA

Pierre Hansen
GERAD and École des Hautes Études Commerciales
Département MQ, 5255 avenue Decelles
Montréal (Québec), H3T 1V6, Canada.

Brigitte Jaumard and Gilles Savard
GERAD and École Polytechnique de Montréal
Département de Mathématiques et de Génie Industriel
C.P. 6079, Succursale "Centre-Ville"
Montréal (Québec), H3C 3A7, Canada.

January 21, 1999

Acknowledgments: Work of the first author was supported by a NSERC fellowship. Work of the second and third authors was supported by FCAR grant #95ER1048. Work of the second author was also supported by NSERC grant #GP0105574. Work of the third author was also supported by NSERC grant #GP0036426, and a FRSQ fellowship. Work of the fourth author was supported by NSERC grant #OGP0046405, and FCAR grant #93ER0141. This paper was also subsidized in part by a grant from CETAI, École des Hautes Études Commerciales and GERAD and a grant from Ultramar Canada, thanks to Luc Massé, as well as by matching funds of NCM_2 of NSERC. We thank an anonymous referee for comments which helped to improve the paper's presentation.

Abstract

We present a branch and cut algorithm that yields in finite time, a globally ϵ-optimal solution (with respect to feasibility and optimality) of the nonconvex quadratically constrained quadratic programming problem. The idea is to estimate all quadratic terms by successive linearizations within a branching tree using Reformulation-Linearization Techniques (RLT). To do so, four classes of linearizations (cuts), depending on one to three parameters, are detailed. For each class, we show how to select the best member with respect to a precise criterion. The cuts introduced at any node of the tree are valid in the whole tree, and not only within the subtree rooted at that node. In order to enhance the computational speed, the structure created at any node of the tree is flexible enough to be used at other nodes. Computational results are reported. Some problems of the literature are solved, for the first time with a proof of global optimality.

Key words: Nonconvex programming, quadratic programming, RLT, linearization, outer-approximation, branch and cut, global optimization.

1 Introduction

The nonconvex quadratically constrained quadratic programming problem (QQP) is a structured global optimization problem, which encompasses many others. Indeed, linear mixed 0-1, fractional, bilinear, bilevel, generalized linear complementarity, and many more programming problems are or can easily be reformulated as particular cases of QQP. This generality has its price: there are theoretical and practical difficulties in the process of solving such problems.

QQP's complexity is present at two levels. The problem of finding a feasible solution is NP-hard as it generalizes the linear complementarity problem (Chung [10] analyzes the complexity of the latter problem); the nonlinear constraints define a feasible region which is in general neither convex nor connected. Moreover, even if the feasible region is a polyhedron, optimizing the quadratic objective function is strongly NP-hard as the resulting problem subsumes the disjoint bilinear programming problem (Hansen, Jaumard and Savard [21] show that an equivalent problem, the linear maxmin problem, is strongly NP-hard). It follows that finding a finite and exact algorithm that solves large QQP's is probably out of reach.

The nonconvex quadratically constrained quadratic programming problem may be stated in its most general form as follows

$$QQP \qquad \begin{array}{ll} \min\limits_{x \in X} & Q^0(x) \\ \text{s.t.} & Q^k(x) \mathrel{\overline{\underline{\leq}}} b_k \qquad k = 1, 2, \ldots, \overline{k}, \end{array}$$

where $X = \{x \in \mathbb{R}^n : \ Ax \leq a\}$, and for each index k in the set $K = \{0, 1, \ldots, \overline{k}\}$

$$\begin{array}{rcl} Q^k : \mathbb{R}^n & \rightarrow & \mathbb{R} \\ x & \mapsto & Q^k(x) = \sum\limits_{(i,j) \in M} C_{ij}^k x_i x_j + \sum\limits_{i \in N} c_i^k x_i^2 + \sum\limits_{i \in N} d_i^k x_i, \end{array}$$

are quadratic functions where $N = \{1, 2, \ldots, n\}$ and $M = \{(i, j) \in N \times N : i > j\}$ are sets of indices. The symbol $\mathrel{\overline{\underline{\leq}}}$ signifies that constraints may be equalities or inequalities. The dimension of the matrices and vectors are the following:

$$\begin{gathered} x \in \mathbb{R}^n; A \in \mathbb{R}^{m \times n}; a \in \mathbb{R}^m; b \in \mathbb{R}^{\overline{k}}; \\ C_{ij}^k, c_i^k, d_i^k \in \mathbb{R} \text{ for all } (i, j) \in M \text{ and } k \in K. \end{gathered}$$

The only further assumptions made in this paper concern the boundedness of the variables. We assume that the constraint $x \geq 0$ is either present in $Ax \leq a$ or implicit through all the constraints. We also suppose that it is possible to obtain valid upper bounds on each variable. This hypothesis is discussed in Section 2.3 below. No

restrictions are imposed regarding convexity or concavity of the objective function or constraints.

In this paper, we develop an algorithm based on approximation of quadratic terms by means of Reformulation-Linearization Techniques (RLT). As surveyed in Section 2.1, such an approach is not new, but is extended here in several ways. First, cuts associated to linearizations are generalized as members of different classes, that depend on one to three parameters. One of them, namely Class $\mathcal{C}_{II}$ defined below, contains a new type of linearization. Second, these classes being defined, we pose and answer the natural question of selecting the best member of each of them under a precise criterion. Third, this outer-approximation scheme is incorporated in the first branch and cut algorithm for QQP. Cuts generated at any node of the exploration tree are valid at all other nodes. Moreover, a key algorithmic element is that the branching structure developed at a node of the tree is reused at several other nodes.

The paper is organized in the following way. The next section introduces linearization of quadratic terms. We present a brief survey of the literature and lay down our assumptions regarding boundedness of the variables. In Section 3, we describe the four classes of valid cuts derived from linearization of quadratic functions. These cuts are used to refine the outer-approximation of quadratic terms, and to eliminate the current relaxed solution. For each class, we show in Section 4 how to select the best cut, i.e., that one which minimizes the worst potential error of the refined approximation. These results lead in Section 5, to a branch and cut algorithm which is shown to converge in finite time within a given tolerance. This final section also details execution of the algorithm on a small example, and reports computational results on a series of problems from the literature. Several of them are solved for the first time with a proof of global optimality.

2 Initial Linearization of Quadratic Terms

The difficulty of QQP lies in the presence of quadratic terms in both objective function and constraints. Throughout this paper, we consider the quadratic functions

$$\begin{array}{rcl} f : \mathbb{R} & \to & \mathbb{R} \\ x_i & \mapsto & f(x_i) = x_i^2 \end{array} \qquad \text{and} \qquad \begin{array}{rcl} g : \mathbb{R}^2 & \to & \mathbb{R} \\ (x_i, x_j) & \mapsto & g(x_i, x_j) = x_i x_j. \end{array}$$

Approximation of the function f is easier than that of g since it is convex on its domain. Any line tangent to f defines a valid under-estimation on the whole domain $\mathbb{R}$. Overestimations are obtained by piecewise linear functions. A more detailed analysis is required for the function g. The plane tangent to g at any given point defines both an

under and over-estimation in different directions. The basic approach described in this paper relies on piecewise estimations of such quadratic functions.

2.1 Survey

The bilinear programming problem (BIL) is equivalent to QQP. The variables of the former problem are partitioned into two sets in such a way that when either set is fixed, the resulting problem has a linear objective function and a polyhedral feasible domain, thus it becomes a linear program. Obviously, BIL is a particular instance of QQP. Reciprocally, any instance of QQP may be reformulated as a BIL by introducing additional variables and constraints. Hansen and Jaumard [19] present various ways of doing so.

In the last few years, several authors studied linearization of quadratic functions. Al-Khayyal and Falk [3] developed an infinitely convergent branch and bound scheme for a problem more general than BIL. The variables of this problem are partitioned into two sets, and require only the three following properties: (i) the objective function is biconvex; (ii) the feasible region is closed and convex; (iii) finite bounds on every variable may be obtained: $x_i \in [\ell_i, u_i]$. Their method relies on outer-approximation of the function $g(x_i, x_j) = x_i x_j$ using the convex envelope over the hyper-rectangle $[\ell_i, u_i] \times [\ell_j, u_j]$. Such a linearization is exact only on the boundary of the hyper-rectangle. If the solution (α_i, α_j) of the corresponding relaxation lies in the strict interior of the hyper-rectangle, then the approximation needs refinement. This is done by adding linearizations over the four sub-intervals $[\ell_i, \alpha_i] \times [\ell_j, \alpha_j]$, $[\alpha_i, u_i] \times [\ell_j, \alpha_j]$, $[\ell_i, \alpha_i] \times [\alpha_j, u_j]$ and $[\alpha_i, u_i] \times [\alpha_j, u_j]$. The branch and bound method generates a new problem for each of these intervals.

Al-Khayyal [1] strengthens this method by also evaluating the concave envelope. Afterwards, Al-Khayyal [2], adapts this idea to BIL by adding linearizations not only to the objective function, but also to the domain. Finally, Al-Khayyal, Larsen and Van Voorhis [4] illustrate a slightly different version of this method on QQP. Instead of generating four new subproblems as above, the proposed method generates only two subproblems by splitting the longest interval in its middle. Computational experiments on randomly generated problems having up to sixteen variables and eight constraints are presented. It appears that the difficulty of a problem is directly related to the number of variables present in quadratic terms that are not at one of their bounds.

Sherali and Alameddine [27] [28] improve the linearization for BIL (where there are no quadratic constraints) by considering the constraints defining the polyhedron X instead of using only the bounding constraints. Sherali and Tuncbilek [29] generalize

this branch and bound method to the case where the functions Q^k are polynomials. Their approach does not consist in reformulating the polynomial problem into a quadratic one by adding new variables and constraints. Instead, they add linearizations of degree higher than two. Sherali and Tuncbilek [30] specialize this method to QP where there are no quadratic constraints. They discuss several improvements such as linearization of cubic terms, diagonalization of the matrix Q^0 (when possible), convex approximation of the function f, and resolution of the relaxation by a Lagrangian method. Ryoo and Sahinidis [26] propose a branch and bound algorithm in which the product $x_i x_j$ is replaced by $\frac{1}{2}(u^2 - x_i^2 - x_j^2)$ where $u = x_i + x_j$. The algorithm solves a sequence of convex underestimating subproblems. Range reduction techniques are used to tighten the bounds on the variables of the subproblems. Sherali and Tuncbilek [31] compare different methods to evaluate bounds for polynomial optimization. Sherali and Tuncbilek [32] present classes of constraints for univariate and multivariate versions of this problem. The branch and bound algorithm uses constraints selection and range reduction strategies.

Generalized Benders decomposition [7] provides a different approach to solve BIL. It is studied in Geoffrion [18], Wolsey [35], Simoes [33], Flippo [13], Floudas and Visweswaran [15], [16], Visweswaran and Floudas [34], and Flippo and Rinnooy Kan [14].

QQP can also be written as a d.c. (difference of convex) programming problem in which the objective function is linear and each function Q^k is expressed as a d.c. function. Phong, Tao and Hoai An [24] presents an outer-approximation method for such problems.

2.2 Initial Relaxation

The classes of cuts associated to quadratic functions presented in Section 3 below lead to outer-approximations of the feasible region. For each i in N, the variable v_i is introduced to estimate the square x_i^2, and for each (i, j) in M, the variable w_{ij} is used to estimate the product $x_i x_j$. Constraints regarding v_i and w_{ij} are successively added to refine the approximation while insuring that the solutions where $v_i = x_i^2$ and $w_{ij} = x_i x_j$ remain feasible.

Let us define precisely the terminology used throughout this paper. The variables v_i and w_{ij} are *estimations* of the quadratic terms x_i^2 and $x_i x_j$. The *linearization of a quadratic function* is obtained by replacing all quadratic terms by their estimations. A *valid inequality* on a given domain is an inequality that does not eliminate any point belonging to both that domain and the feasible region. When valid inequalities

are combined, the resulting feasible region is an outer-approximation of the original domain. Solution of this relaxed problem yields the *current point*. A *cut* is a valid inequality that eliminates the current point. In Section 3, specific cuts derived from linearization of a quadratic functions are called *linearizations*.

We use the RLT notation introduced by Sherali and Tuncbilek [29]; the linearization described above, in which the quadratic terms are replaced by linear ones, is denoted by $[\cdot]_\ell$. Typically, $[(a_p - A_{p.}x)(a_q - A_{q.}x)]_\ell$ (where $A_{p.}$ is the p^{th} row of the matrix A) denotes the linearization of the product

$$(a_p - A_{p.}x)(a_q - A_{q.}x) \;=\; \sum_{i\in N}\sum_{j\in N} A_{pi}A_{qj}x_ix_j - \sum_{i\in N}(a_pA_{qi} + a_qA_{pi})x_i + a_pa_q,$$

and is explicitly written through introduction of the linearization variables v and w

$$\sum_{(i,j)\in M}(A_{pi}A_{qj} + A_{pj}A_{qi})w_{ij} + \sum_{i\in N} A_{pi}A_{qi}v_i - \sum_{i\in N}(a_pA_{qi} + a_qA_{pi})x_i + a_pa_q.$$

Since the definition of the polyhedron X contains the constraints $A_{p.}x \le a_p$ and $A_{q.}x \le a_q$, one can obtain a valid inequality in a higher dimensional space by imposing linearization of the product to be greater than or equal to zero. We are now able to formulate an initial linear relaxation of QQP, which is a RLT relaxation as in Sherali and Tuncbilek [29] [31] [32].

Proposition 2.1 *Let P be a subset of indices of $\{1, 2, \ldots, m\}$, and Q_p a subset of $\{p, p+1, \ldots, m\}$ for each p of P. The problem*

$$[QQP]_\ell \qquad \begin{array}{lll} \min\limits_{x\in X, v, w} & [Q^0(x)]_\ell & \\ s.t. & [Q^k(x)]_\ell \mathrel{\overline{=}} b_k & k = 1, 2, \ldots, \overline{k}, \\ & [(a_p - A_{p.}x)(a_q - A_{q.}x)]_\ell \ge 0 & p \in P,\ q \in Q_p \end{array}$$

is a linear relaxation of QQP.

Proof: Let x^* be an optimal solution of QQP. Set $v_i^* = x_i^{*2}$ for each i belonging to N, and $w_{ij}^* = x_i^*x_j^*$ for each (i,j) in M.

For each index k of K, the value $[Q^k(x^*)]_\ell = \sum_{(i,j)\in M} C_{ij}^k w_{ij}^* + \sum_{i\in N} c_i^k v_i^* + \sum_{i\in N} d_i^k x_i^*$, is identical to $Q^k(x^*)$. Moreover, by replacing the variables v_i^* and w_{ij}^* by the respective products x_i^{*2} and $x_i^*x_j^*$, the linearization constraints become $(a_p - A_{p.}x^*)(a_q - A_{q.}x^*) \ge 0$ for $p \in P$ and $q \in Q_p$. The point (x^*, v^*, w^*) is feasible for the problem $[QQP]_\ell$ and its objective value is equal to $Q^0(x^*)$, thus the result follows. ∎

In the case where all possible valid inequalities derived from the linearizations are present in the relaxation, i.e., when $P = \{1, 2, \ldots, m\}$, and $Q_p = \{p, p+1, \ldots, m\}$

for each p of P, Sherali and Tuncbilek [30] show that if at least one of the variables x_i is lower and upper bounded in X, then the constraint $x \in X$ is redundant for that linearization. It can therefore be withdrawn from $[QQP]_\ell$.

This linearization technique provides a general framework for an outer-approximation method that consists essentially in a sequence of refinements of approximations of quadratic functions.

2.3 Computation of Bounds on Each Variable

Due to the nonconvex nature of the constraints of QQP, obtaining tight bounds on the variables is a nontrivial problem. The range reduction strategy that we use is that of [32]. Let x^-, x^+ be bounds on x_i such that $0 \le x^- \le x_i \le x^+$, and v^-, v^+ be bounds on v_i such that $0 \le v^- \le v_i \le v^+$ obtained by replacing $[QQP]_\ell$'s objective function by $\pm x_i$ and then by $\pm v_i$. Let $\ell_i \equiv \max\{x^-, \sqrt{v^-}\}$ and $u_i \equiv \min\{x^+, \sqrt{v^+}\}$. If $\ell_i \le u_i$, then ℓ_i and u_i are valid bounds for the variable x_i over the feasible domain of QQP. If $\ell_i > u_i$, then the feasible domain of QQP is empty.

These bounds may be improved if $\ell_i > \min\{x_i : x \in X\}$ or if $u_i < \max\{x_i : x \in X\}$. After adding the bounding constraints $\ell_i \le x_i \le u_i$ to the polyhedron X, one can reiterate the process in order to tighten the interval. Consider the following example.

Example 2.2 *Let a feasible domain be defined by the two constraints*

$$x_1 + x_1^2 \le 6 \quad \text{and} \quad x_1 \ge 1.$$

One can easily verify that the feasible interval is $[1, 2]$. *The approach described above yields the constraints of the relaxation* $[QQP]_\ell$*:*

$$x_1 + v_1 \le 6,\ x_1 \ge 1 \quad \text{and} \quad [(x_1 - 1)^2]_\ell = v_1 - 2x_1 + 1 \ge 0.$$

The first computed bounds are $x_1 \in [1, \frac{7}{3}]$ *and* $v_1 \in [1, 5]$, *from which it follows that* $x_1 \in [1, \sqrt{5}]$. *Since* $X = \{x : x_1 \ge 1\}$ *has no upper bound, it is possible that the new upper bound* $\sqrt{5}$ *can be improved. By adding the constraint* $x_1 \le \sqrt{5}$ *to* X, *we obtain the new linearizations*

$$\begin{aligned} [(\sqrt{5} - x_1)^2]_\ell &= v_1 - 2\sqrt{5}x_1 + 5 \ge 0 \quad \text{and} \\ [(\sqrt{5} - x_1)(x_1 - 1)]_\ell &= -v_1 + (1 + \sqrt{5})x_1 - \sqrt{5} \ge 0. \end{aligned}$$

Once again, these constraints generate new bounds: $x_1 \in [1, \frac{11}{2\sqrt{5}+1}]$ *and* $v_1 \in [1, 13 - 4\sqrt{5}]$. *Since* $\frac{11}{2\sqrt{5}+1} \approx 2.010 < \sqrt{13 - 4\sqrt{5}} \approx 2.014 < \sqrt{5} \approx 2.236$, *the constraint* $x_1 \le \sqrt{5}$ *can be improved to* $x_1 \le \frac{11}{2\sqrt{5}+1}$. *Repetition of this process converges to the feasible interval of* x_1, *i.e., to* $[1, 2]$. ∎

Any valid inequality belonging to the two first linearization classes presented in the following section can also be added when evaluating bounds. It is then possible that better bounds are generated. In Section 5, we present an algorithm whose pre-processing phase consists in iterating this bounding process until improvement is negligible. The only assumption made in this paper concerning QQP is that finite bounds may be obtained in this way for every variable.

3 Classes of Linearizations

In this section, we present four classes of linearizations. Each class consists of a set of inequalities which are valid over the intervals $[\ell_i, u_i]$ for $i \in N$.

The first class of linearizations, due to Al-Khayyal and Falk [3], contains under-estimations of the square function. For $\alpha_i \in [\ell_i, u_i], i \in N$ consider the RLT constraints

$$\underline{V}_i(\alpha_i): \qquad [(x_i - \alpha_i)^2]_\ell \geq 0.$$

For a given value of α_i, the valid inequality defines the half-space tangent to the convex function x_i at the point $x_i = \alpha_i$. The first class of valid inequalities is

$$\mathcal{C}_I \;=\; \left\{\underline{V}_i(\alpha_i):\ \alpha_i \in [\ell_i, u_i], i \in N\right\}.$$

The second class of linearizations, which is new, contains under-estimations of a paraboloid. For $\alpha_i \in [\ell_i, u_i], \alpha_j \in [\ell_j, u_j], i, j \in N, \gamma \in \mathbb{R}$ consider the RLT constraints

$$\underline{P}^{\gamma}_{ij}(\alpha_i, \alpha_j): \qquad [((\alpha_i - x_i) + \gamma(\alpha_j - x_j))^2]_\ell \geq 0.$$

For given values of α_i, α_j and γ, the valid inequality defines the half-space tangent to the convex paraboloid $(x_i + \gamma x^j)^2$ at the point $(x_i, x_j) = (\alpha_i, \alpha_j)$. The second class of valid inequalities is

$$\mathcal{C}_{II} \;=\; \left\{\underline{P}^{\gamma}_{ij}(\alpha_i, \alpha_j):\ \alpha_i \in [\ell_i, u_i], \alpha_j \in [\ell_j, u_j], i, j \in N, \gamma \in \mathbb{R}\right\}.$$

Both the inequalities of the above classes are tangent linear under-estimations of convex functions, and thus are valid everywhere. The inequalities of the next two classes are not. Dichotomy is used to refine the approximations of the quadratic terms on subintervals. The branch and cut algorithm introduces some variables $\delta_i(\alpha_i) \in [0, 1]$ where $\alpha_i \in [\ell_i, u_i]$ for $i \in N$. The branching process of the algorithm fixes these variables to either 0 or 1. When backtracking, the variables are freed in $[0, 1]$. The same variable $\delta_i(\alpha_i)$ can be used at different branches of the enumeration tree.

The branch and cut algorithm presented in Section 5 uses a best first strategy. When the algorithm unfolds, it often jumps from one branch to another. The RLT inequalities valid at one branch may not be valid at another. Instead of adding and deleting inequalities when moving along the branches, the introduction of the variables δ ensures that the inequalities are valid everywhere.

The constraints of classes III and IV are constructed in such a way that they are active only when the variables are fixed at either 0 or 1. When these are free, the constraints are redundant. The variables $\delta_i(\ell_i)$ and $\delta_i(u_i)$ are respectively fixed to 1 and 0. Moreover, if $\ell_i \leq \alpha_i \leq \beta_i \leq u_i$ then $\delta_i(\beta_i) \geq \delta_i(\alpha_i)$. Therefore, if $\delta_i(\alpha_i) = 0$ then $\delta_i(\beta_i) = 0$, and if $\delta_i(\beta_i) = 1$ then $\delta_i(\alpha_i) = 1$.

The third class of linearizations are over-estimations of the square function. For $\alpha_i, \beta_i \in [\ell_i, u_i], \alpha_i < \beta_i, i \in N$ consider the constraints

$$\overline{V}_i(\alpha_i, \beta_i): \qquad [(\alpha_i - x_i)(\beta_i - x_i)]_\ell \geq (u_i - \ell_i)^2(\delta_i(\alpha_i) - \delta_i(\beta_i) - 1).$$

For given values of α_i and β_i, the valid inequality defines the half-space obtained through the cord from $(x_i, v_i) = (\alpha_i, \alpha_i^2)$ to (β_i, β_i^2). If $\delta_i(\alpha_i) = 1$ and $\delta_i(\beta_i) = 0$, then the inequality reduces to $[(\alpha_i - x_i)(\beta_i - x_i)]_\ell \geq 0$, and is thus valid when $\alpha_i \leq x_i \leq \beta_i$. Otherwise, the right-hand-side of the inequality becomes $-(u_i - \ell_i)^2$, and thus the inequality remains valid when $\ell_i \leq x_i < \alpha_i$ or $\beta_i < x_i \leq u_i$. The second class of valid inequalities is

$$\mathcal{C}_{III} \;=\; \left\{\underline{V}_i(\alpha_i), \underline{V}_i(\beta_i), \overline{V}_i(\alpha_i, \beta_i): \; \alpha_i, \beta_i \in [\ell_i, u_i], \alpha_i < \beta_i, i \in N\right\}.$$

By combining $\overline{V}_i(\alpha_i, \beta_i)$ with $\underline{V}_i(\alpha_i)$ and $\underline{V}_i(\beta_i)$, it follows that if $\delta_i(\alpha_i) = 1$ and $\delta_i(\beta_i) = 0$ then $\alpha_i \leq x_i \leq \beta_i$, thus making $\overline{V}_i(\alpha_i, \beta_i)$ the concave hull of $x_i x_j$ on that subinterval.

The fourth class of linearizations are estimations of the product of two variables.

Consider the tangent plane Π to the function $g(x_i, x_j) = x_i x_j$ at the point (α_i, α_j). This plane satisfies the three following properties:

i- (x_i, x_j, w_{ij}) belongs to Π if and only if $x_i = \alpha_i$ or $x_j = \alpha_j$.

ii- Π strictly under-estimates (x_i, x_j, w_{ij}) if and only if both $x_i < \alpha_i$ and $x_j < \alpha_j$, or both $x_i > \alpha_i$ and $x_j > \alpha_j$.

iii- Π strictly over-estimates (x_i, x_j, w_{ij}) if and only if both $x_i < \alpha_i$ and $x_j > \alpha_j$, or both $x_i > \alpha_i$ and $x_j < \alpha_j$.

We define the four quadrants associated to the point (α_i, α_j):

$$\begin{array}{rclcrcl} \Omega^I & = & \{(x_i,x_j) : x_i \geq \alpha_i, x_j \geq \alpha_j\}, & & \Omega^{II} & = & \{(x_i,x_j) : x_i \leq \alpha_i, x_j \geq \alpha_j\}, \\ \Omega^{III} & = & \{(x_i,x_j) : x_i \leq \alpha_i, x_j \leq \alpha_j\}, & & \Omega^{IV} & = & \{(x_i,x_j) : x_i \geq \alpha_i, x_j \leq \alpha_j\}. \end{array}$$

For $\alpha_i \in [\ell_i, u_i], \alpha_j \in [\ell_j, u_j], i, j \in N$, set $L_i = \alpha_i - \ell_i, L_j = \alpha_j - \ell_j$, and $U_i = u_i - \alpha_i, U_j = u_j - \alpha_j$ and consider the constraints

$$\begin{array}{ll} \underline{W}^I_{ij}(\alpha_i,\alpha_j): & [(x_i-\alpha_i)(x_j-\alpha_j)]_\ell \geq L_iU_j(\delta_i(\alpha_i)-1) + U_iL_j(\delta_j(\alpha_j)-1), \\ \overline{W}^{II}_{ij}(\alpha_i,\alpha_j): & [(x_i-\alpha_i)(x_j-\alpha_j)]_\ell \leq U_iU_j\delta_i(\alpha_i) + L_iL_j(1-\delta_j(\alpha_j)), \\ \underline{W}^{III}_{ij}(\alpha_i,\alpha_j): & [(x_i-\alpha_i)(x_j-\alpha_j)]_\ell \geq -U_iL_j\delta_i(\alpha_i) - L_iU_j\delta_j(\alpha_j), \\ \overline{W}^{IV}_{ij}(\alpha_i,\alpha_j): & [(x_i-\alpha_i)(x_j-\alpha_j)]_\ell \leq L_iL_j(1-\delta_i(\alpha_i)) + U_iU_j\delta_j(\alpha_j). \end{array}$$

For given values of α_i and α_j, the valid inequalities $\underline{W}^I_{ij}(\alpha_i,\alpha_j)$ and $\underline{W}^{III}_{ij}(\alpha_i,\alpha_j)$ define the convex hulls of the function x_ix_j on the respective domains Ω^I and Ω^{III}. Indeed, on their respective domains the right-hand-sides of these inequalities becomes 0, thus the inequalities are those of Al-Khayyal and Falk [3]. Similarly, the valid inequalities $\underline{W}^{II}_{ij}(\alpha_i,\alpha_j)$ and $\underline{W}^{IV}_{ij}(\alpha_i,\alpha_j)$ define the concave hulls of the function x_ix_j on the respective domains Ω^{II} and Ω^{IV}. The fourth class of valid inequalities is partitioned into

$$\begin{array}{rcl} \underline{\mathcal{C}}_{IV} & = & \{\ \underline{W}^I_{ij}(\alpha_i,\alpha_j),\ \underline{W}^{III}_{ij}(\alpha_i,\alpha_j) :\ \alpha_i \in [\ell_i,u_i], \alpha_j \in [\ell_j,u_j], i,j \in N\ \} \\ \overline{\mathcal{C}}_{IV} & = & \{\ \overline{W}^{II}_{ij}(\alpha_i,\alpha_j),\ \overline{W}^{IV}_{ij}(\alpha_i,\alpha_j) :\ \alpha_i \in [\ell_i,u_i], \alpha_j \in [\ell_j,u_j], i,j \in N\ \}. \end{array}$$

4 Selection of the Best Linearization

In this section, we study how to find among the four classes of cuts presented above, that one which should be added to the relaxation in order to obtain the best possible estimation of quadratic terms, according to a precise criterion. In a branching algorithm, selecting the branching value in such a way may reduce the number of nodes generated by the algorithm. This is illustrated in Section 5.5.

4.1 Refinement of Under-Estimation of the Square Function

We consider in this section the case where the current point satisfies $\hat{v}_i < \hat{x}_i^2$. The question studied here consists in finding the best linearization among all $\underline{V}_i(\alpha)$ that eliminate the current point. The point $(\hat{x}_i, \hat{v}_i)$ is obtained by solving a relaxation of

QQP. The fact that $\hat{x}_i^2$ is not well-approximated by $\hat{v}_i$ can be interpreted in two ways: either the value $\hat{v}_i$ is too small, or that of $\hat{x}_i$ is too large. Therefore, we propose to select the point α (and thus the linearization $\underline{V}_i(\alpha)$) that minimizes a potential error over the interval $[\sqrt{\hat{v}_i}, \hat{x}_i]$.

The error to minimize is the largest value between $e_1 = \hat{v}_i - (2\alpha\sqrt{\hat{v}_i} - \alpha^2)$ and αe_2 where $e_2 = (\frac{\hat{x}_i^2}{2\alpha} + \frac{\alpha}{2}) - \hat{x}_i$. The weight of the second error term α compensates for the fact that e_1 represents a difference of squared values, but not e_2. Figure 1 illustrates these error terms as well as the linearization $\underline{V}_i(\alpha)$ tangent to the curve x_i^2.

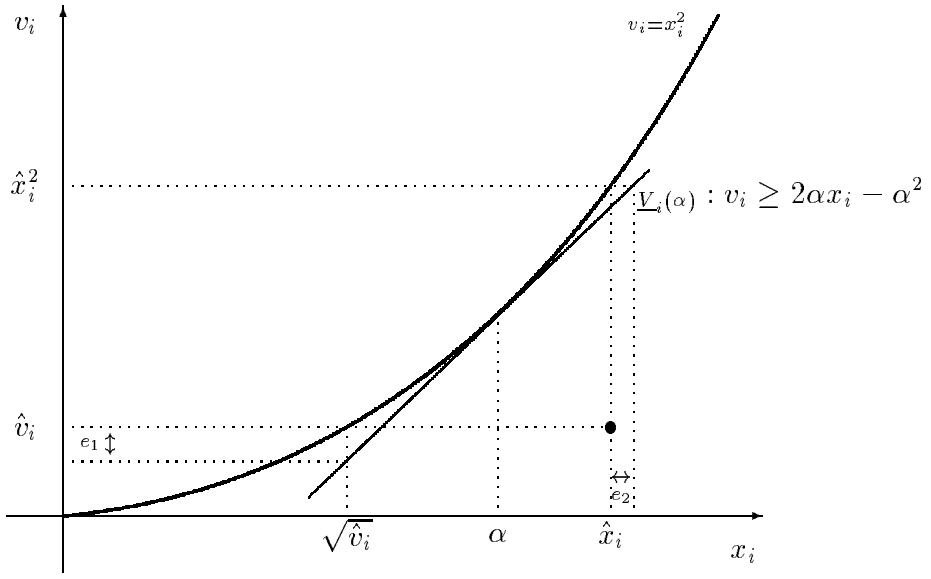

Figure 1: Minimization of the error for under-estimating a square

Slight variations of the following observation are used when minimizing various weighted errors: the least value of the maximum of e_1 and αe_2 is attained when $e_1 = \alpha e_2$, since e_1 monotonically increases from 0 and αe_2 monotonically decreases to 0 when α varies continuously from $\sqrt{\hat{v}_i}$ to $\hat{x}_i$.

Proposition 4.1 *The value* $\alpha = (\sqrt{2} - 1)\hat{x}_i + (2 - \sqrt{2})\sqrt{\hat{v}_i} \in [\sqrt{v_i}, x_i]$ *minimizes the maximum between* $e_1 = \hat{v}_i - (2\alpha\sqrt{\hat{v}_i} - \alpha^2)$ *and* αe_2 *where* $e_2 = (\frac{\hat{x}_i^2}{2\alpha} + \frac{\alpha}{2}) - \hat{x}_i$.

Proof: The least value of the maximum between e_1 and αe_2 is attained when both terms are equal. One can easily verify that $e_1 = \alpha e_2$ if and only if

$$\alpha^2 + 2(\hat{x}_i - 2\sqrt{\hat{v}_i})\alpha + 2\hat{v}_i - \hat{x}_i^2 = 0.$$

The root of this quadratic function that lies in the interval $[\sqrt{v_i}, x_i]$ is a convex combination of the endpoints $\alpha = (2 - \sqrt{2})\sqrt{\hat{v}_i} + (\sqrt{2} - 1)\hat{x}_i$. ∎

4.2 Refinement of Under-Estimation of a Paraboloid

The inequality $\underline{P}_{ij}^{\gamma}(\alpha_i, \alpha_j)$ depends on the three parameters α_i, α_j and γ. The firt two of them define the point where the tangent plane is evaluated, the third parameter, γ, describes the curvature of the paraboloid. In order to select its value, we rewrite the valid inequality:

$$(v_j - 2\alpha_j x_j + \alpha_j^2)\gamma^2 + 2(w_{ij} - \alpha_i x_j - \alpha_j x_i + \alpha_i \alpha_j)\gamma + (v_i - 2\alpha_i x_i - \alpha_i^2) \geq 0. \qquad (1)$$

In the case where the left-hand-side is convex with respect to γ, we choose the value of γ, specified in the next proposition, that corresponds to the minimum of this quadratic function. The case where it is concave is not developed since it reduces to the use of an arbitrarily large γ, and the linearizations become equivalent to those of class $\mathcal{C}_I$.

In order to lighten the notation, we write

$$\tau_i \equiv \hat{v}_i - 2\alpha_i \hat{x}_i + \alpha_i^2, \quad \tau_j \equiv \hat{v}_j - 2\alpha_j \hat{x}_j + \alpha_j^2, \quad \pi_{ij} \equiv \hat{w}_{ij} - \alpha_i \hat{x}_j - \alpha_j \hat{x}_i + \alpha_i \alpha_j.$$

Proposition 4.2 *If $\ell_i \leq \hat{x}_i < \sqrt{\hat{v}_i} \leq u_i$ and $\ell_j \leq \hat{x}_j < \sqrt{\hat{v}_j} \leq u_j$ and if $\tau_i \tau_j < \pi_{ij}^2$, then for any $\alpha_i \in]\hat{x}_i, \sqrt{\hat{v}_i}[$ and $\alpha_j \in]\hat{x}_j, \sqrt{\hat{v}_j}[$ the value of γ for which the cut $\underline{P}_{ij}^{\gamma}(\alpha_i, \alpha_j)$ is the deepest at the current point (i.e., the value of $[((\alpha_i - \hat{x}_i) + \gamma(\alpha_j - \hat{x}_j))^2]_\ell$ is minimal) is $\gamma = -\frac{\pi_{ij}}{\tau_j}$, and it eliminates the current point.*

Proof: At the current point $(\hat{x}_i, \hat{x}_j, \hat{v}_i, \hat{v}_j, \hat{w}_{ij})$, the value of the left-hand-side of $\underline{P}_{ij}^{\gamma}(\alpha_i, \alpha_j)$ becomes $\tau_j \gamma^2 + 2\pi_{ij}\gamma + \tau_i$. It is convex with respect to γ since $\tau_j > \hat{x}_j^2 - 2\alpha_j \hat{x}_j + \alpha_j^2 \geq 0$, and its minimum is attained when $\gamma = -\frac{\pi_{ij}}{\tau_j}$ (this choice of γ maximizes in that way the depth of the inequality). It can be written

$$\frac{\pi_{ij}^2}{\tau_j} - \frac{2\pi_{ij}^2}{\tau_j} + \tau_i \geq 0 \qquad \text{or} \qquad \pi_{ij}^2 \leq \tau_i \tau_j.$$

Since $\tau_i \tau_j < \pi_{ij}^2$, the current point is eliminated. ∎

We now show how to obtain the point where the tangent plane is evaluated, i.e., the values of α_i and α_j that minimize the greatest potential error are selected.

Proposition 4.3 *The value $(\alpha_i, \alpha_j) = (\hat{x}_i + \sqrt{\hat{v}_i}, \hat{x}_j + \sqrt{\hat{v}_j})/2 \in [x_i, \sqrt{v_i}] \times [x_j, \sqrt{v_j}]$ minimizes the maximum between the distances from the paraboloid to the tangent plane at the point $(\hat{x}_i, \sqrt{\hat{v}_j})$: $e_1 = h_\gamma(\hat{x}_i, \sqrt{\hat{v}_j}) - \tau(\hat{x}_i + \gamma\sqrt{\hat{v}_j}) + \tau^2/2$ and at the point $(\sqrt{\hat{v}_i}, \hat{x}_j)$: $e_2 = h_\gamma(\sqrt{\hat{v}_i}, \hat{x}_j) - \tau(\sqrt{\hat{v}_i} + \gamma\hat{x}_j) + \tau^2/2$, where $\tau = 2(\alpha_i + \gamma\alpha_j)$.*

Proof: The minimal value of the maximum between e_1 and e_2 is attained when both terms are equal. One can easily verify that $e_1 = e_2$ if and only if

$$\tau = (\hat{x}_i + \sqrt{\hat{v}_i}) + \gamma(\hat{x}_j + \sqrt{\hat{v}_j}).$$

Since $\tau = 2(\alpha_i + \gamma\alpha_j)$, the values $\alpha_i = (\hat{x}_i + \sqrt{\hat{v}_i})/2$ and $\alpha_j = (\hat{x}_j + \sqrt{\hat{v}_j})/2$ satisfy the criterion. ∎

Similarly, it can be shown that the same point minimizes the maximal distance between the paraboloid and the tangent plane at the points $(\hat{x}_i, \hat{x}_j)$ and $(\sqrt{\hat{v}_i}, \sqrt{\hat{v}_j})$.

4.3 Refinement of Over-Estimation of the Square Function

Suppose now that we wish to eliminate an over-approximation of the function f, when the current point satisfies $\hat{v}_i > \hat{x}_i^2$. We discuss the choice of the value α that minimizes the greatest potential weighted error. The choice of this parameter is more delicate than that of the under-estimation since a linearization at a point depends on two other parameters: the values of both inferior and superior evaluation points. Let $\underline{\beta}$ and $\overline{\beta}$ be such evaluation points satisfying $[\hat{x}_i, \sqrt{\hat{v}_i}] \subset]\underline{\beta}, \overline{\beta}[$. We wish to refine the approximation on this first interval.

Proposition 4.4 *The value* $\alpha = \frac{-q+\sqrt{q^2+4p\overline{\beta}(\hat{x}_i-\underline{\beta})\hat{x}_i}}{2p} \in [x_i, \sqrt{v_i}]$ *where* $p = (\hat{x}_i - \underline{\beta}) + (\overline{\beta} - \sqrt{\hat{v}_i})$ *and* $q = (\hat{x}_i - \underline{\beta})(\overline{\beta} - \hat{x}_i) - (\overline{\beta} - \sqrt{\hat{v}_i})\sqrt{\hat{v}_i}$, *minimizes the maximum between* $e_1 = (\underline{\beta} + \alpha)\hat{x}_i - \underline{\beta}\alpha - \hat{x}_i^2$ *and* αe_2 *where* $e_2 = \sqrt{\hat{v}_i} - \frac{\hat{v}_i + \alpha\overline{\beta}}{\alpha + \overline{\beta}}$.

Proof: The least value of the maximum between e_1 and αe_2 is attained when both terms are equal. It can be shown that $e_1 = \alpha e_2$ if and only if

$$\begin{aligned}
(\underline{\beta} + \alpha)(\alpha + \overline{\beta})\hat{x}_i - \underline{\beta}\alpha(\alpha + \overline{\beta}) - \hat{x}_i^2(\alpha + \overline{\beta}) &= \sqrt{\hat{v}_i}(\alpha + \overline{\beta})\alpha - (\hat{v}_i + \alpha\overline{\beta})\alpha \\
&\Longleftrightarrow \\
(\hat{x}_i - \underline{\beta} + \overline{\beta} - \sqrt{\hat{v}_i})\alpha^2 + ((\hat{x}_i - \underline{\beta})(\overline{\beta} - \hat{x}_i) - (\overline{\beta} - \sqrt{\hat{v}_i})\sqrt{\hat{v}_i})\alpha - (\hat{x}_i - \underline{\beta})\overline{\beta}\hat{x}_i &= 0 \\
&\Longleftrightarrow \\
p\alpha^2 + q\alpha - (\hat{x}_i - \underline{\beta})\overline{\beta}\hat{x}_i &= 0.
\end{aligned}$$

This convex quadratic equation necessarily possesses a root in the interval $[\hat{x}_i, \sqrt{\hat{v}_i}]$. Indeed, if $\alpha = \hat{x}_i$ then $e_1 = 0$ and $\alpha e_2 > 0$, then by continuously increasing α up to $\sqrt{\hat{v}_i}$, e_1 becomes and remains positive, and αe_2 decreases to 0. Therefore, there exists a value of α in the interval such that $e_1 = \alpha e_2$.

The second root is negative. Indeed, if $\alpha = 0$ then $e_1 < 0$ and $\alpha e_2 = 0$, thus by continuously decreasing α, the value of αe_2 decreases more rapidly than that of e_1, and so there exists an $\alpha < 0$ such that $e_1 = \alpha e_2$. Therefore, the desired root is the positive one. ∎

4.4 Refinement of the Product of Two Variables

We finally consider the case where we wish to refine the estimation of the product of two variables x_i and x_j. We only cover explicitly the case of the over-approximation since the under-estimation is symmetrical.

Let $\hat{w}_{ij}$ be an estimation of $\hat{x}_i\hat{x}_j$ such that $\hat{w}_{ij} > \hat{x}_i\hat{x}_j$, and $\underline{\beta}$ and $\overline{\beta}$ the evaluation points such that $[\hat{x}_j, \frac{\hat{w}_{ij}}{\hat{x}_i}] \subset]\underline{\beta}, \overline{\beta}[$. We wish to obtain an evaluation point $\alpha \in [\hat{x}_i, \frac{\hat{w}_{ij}}{\hat{x}_j}]$ for the variable x_i, in order to refine the approximation of the product x_ix_j when $(x_i, x_j) \in [\hat{x}_i, \frac{\hat{w}_{ij}}{\hat{x}_j}] \times [\underline{\beta}, \overline{\beta}]$ to eliminate the over-estimation. Since in the algorithm presented in Section 5 branching is done on a single variable, we fix variable x_j to $\alpha_j = (\hat{x}_j + \sqrt{\hat{v}_j})/2$. Therefore, we select the value of α that minimizes the greatest potential weighted error in the interval $[\hat{x}_i, \frac{w_{ij}}{\hat{x}_j}[$.

Proposition 4.5 *The value*

$$\alpha = \frac{(1 - \frac{\overline{\beta}}{\alpha_j})\hat{x}_i + (1 - \frac{\alpha_j}{\underline{\beta}})\frac{\hat{w}_{ij}}{\hat{x}_j}}{(1 - \frac{\overline{\beta}}{\alpha_j}) + (1 - \frac{\alpha_j}{\underline{\beta}})}$$

minimizes the maximum between $e_1 = \overline{\beta}\hat{x}_i + \alpha(\alpha_j - \overline{\beta}) - \alpha_j\hat{x}_i$ *and* $\alpha_j e_2$, *where* $e_2 = \frac{\hat{w}_{ij}}{\hat{x}_j} - (\frac{\alpha_j\hat{w}_{ij}}{\hat{x}_j\underline{\beta}} - \frac{\alpha(\alpha_j - \underline{\beta})}{\underline{\beta}})$.

Proof: The least value of the maximum between e_1 and $\alpha_j e_2$ is attained when both terms are equal. Figure 2 illustrates these error terms. One can verify that $e_1 = \alpha_j e_2$ if and only if

$$\alpha(\alpha_j - \overline{\beta} - \frac{\alpha_j(\alpha_j - \underline{\beta})}{\underline{\beta}}) = -\overline{\beta}\hat{x}_i + \alpha_j(\hat{x}_i + \frac{\hat{w}_{ij}}{\hat{x}_j} - \frac{\alpha_j\hat{w}_{ij}}{\hat{x}_j\underline{\beta}})$$

The results follows by solving for the variable α. ∎

Similar results can be obtained for the case of under-approximation of the product of two variables.

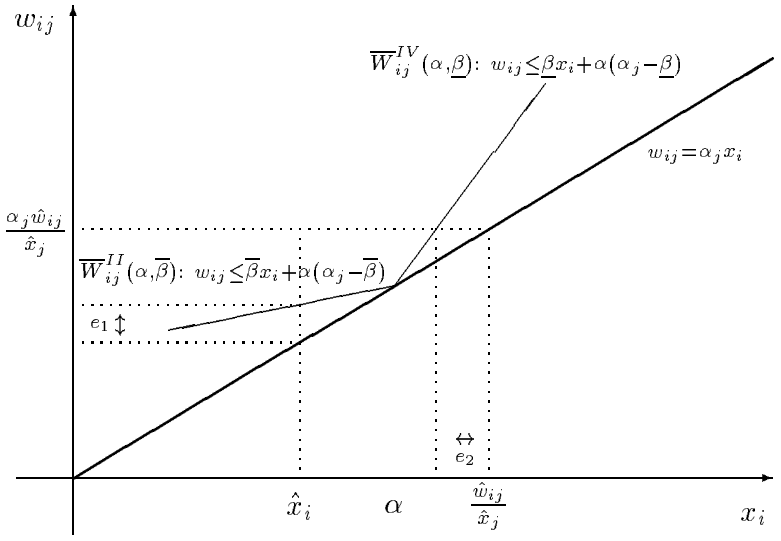

Figure 2: Minimization of the error for over-estimating a product

5 Branch and Cut Algorithm

In this section, we use the linearizations described above in order to construct a solution method for QQP. We seek an approximate solution in terms of both feasibility and objective function value.

Definition 5.1 *For a feasibility tolerance parameter $\epsilon_r > 0$, a solution $(\hat{x}, \hat{v}, \hat{w})$ is said to be* ϵ_r-approximate *if it satisfies the three following conditions: (i) $\hat{x} \in X$;*
(ii) $|\hat{x}_i^2 - \hat{v}_i| < \epsilon_r$ for each i of N; (iii) $|\hat{x}_i \hat{x}_j - \hat{w}_{ij}| < \epsilon_r$ for each (i,j) of M.
Moreover, let z^* be the minimum value of the linearization $[Q^0]_\ell$ over all ϵ_r-approximate solutions. For an objective tolerance $\epsilon_z > 0$, the solution is said to be ϵ_r-ϵ_z-optimal *if it is ϵ_r-approximate and if $z^* - [Q^0(\hat{x}, \hat{v}, \hat{w})]_\ell < \epsilon_z$.* ∎

The following notation is used throughout this section. The branch and cut algorithm generates a branching *tree*. The initial node of the tree is called the *root*. When a node is processed and needs further refinement, the branching step creates two new nodes: these are called *sons* of the *father*. The *incumbent solution* refers to the best solution currently found by the algorithm. Its objective value is the *incumbent value* which is set to $+\infty$ at the beginning of the execution.

This section is divided into five parts. We first describe the pre-processing step done on the instance to obtain valid bounds on the variables, and the initialization steps at the root of the tree. Second, we discuss the branching strategy that selects

at each node a branching variable and a branching value. Third, we detail the cutting step which refines the outer-approximation. Fourth, all these steps are combined in a branch and cut algorithm. Finally, the method is illustrated on a small example and computational results on problems of the literature are reported.

5.1 Root Node

At the root node, the linear problem $[QQP]_\ell$ is created and contains a number of linearizations. Next, bounds are evaluated for each variable through the iterative process described in Section 2.3. Then, the outer-approximation of quadratic terms is refined using the linearizations presented in Section 3, without however adding any variables δ_i. In this process, only cuts associated to convex and concave envelopes are introduced.

The first phase of the algorithm, the *pre-processing*, consists of the iteration of these steps until no bound is improved by more than ϵ_r. This relaxation tightening is performed only at the root node. Although repeating this at other nodes might be useful, it would change the algorithm from a branch and cut one into a branch and bound one (and thus some cuts would be valid only in parts of the tree).

After the pre-processing phase at the root node, further refinement of the outer-approximation requires introduction of cuts from class $\mathcal{C}_{III}$. The algorithm moves on to the branching process.

5.2 Branching

When the need for refining the outer-approximation is felt at a node of the enumeration tree, the branching step of the algorithm is invoked. Then, branching variable and branching value are selected in the following way, by considering the optimal solution $(\hat{x}, \hat{v}, \hat{w})$ of the current linear relaxation.

Selection of the branching variable x_i:
Select i in $\text{argmax}_i\{|\hat{v}_i - \hat{x}_i^2|\}$, and (k,j) in $\text{argmax}_{(k,j)}\{|\hat{w}_{kj} - \hat{x}_k\hat{x}_j|\}$. If the error associated to the indices (k,j) is larger than that of the index i, then reset i to be the index corresponding to the largest error between $|\hat{v}_k - \hat{x}_k^2|$ and $|\hat{v}_j - \hat{x}_j^2|$. The selected branching variable is x_i. ∎

If the error (either $|\hat{v}_i - \hat{x}_i^2|$ or $|\hat{w}_{kj} - \hat{x}_k\hat{x}_j|$) associated to the branching variable is less than ϵ_r, then the branching step does not need to be pursued as the corresponding solution is ϵ_r-feasible. Otherwise, a branching value must be evaluated.

At various nodes of the tree, branching structures have been developed for different values of x_i. Let $\underline{\beta}$ be the greatest branching value less than $\hat{x}_i$ currently produced by the algorithm. If there are none, set $\underline{\beta}$ to ℓ_i. Similarly, let $\overline{\beta}$ be the smallest branching value greater than $\hat{x}_i$ currently produced by the algorithm. If there are none, set $\overline{\beta}$ to u_i. It is possible that the branching done to reach the current node from the root does not require fixation of $\delta_i(\underline{\beta})$ or of $\delta_i(\overline{\beta})$, i.e., at least one of these variables is free. In that case, the branching step selects that branching value (and so, the whole branching structure already present in the model is reused). Otherwise, the branching value is chosen according to the results presented in Section 4. This process is now described in algorithmic terms.

Selection of the branching value α:
One of the four following cases occurs.

i- Both $\delta_i(\underline{\beta})$ and $\delta_i(\overline{\beta})$ are free in $[0,1]$: Set α to be the value among $\underline{\beta}$ and $\overline{\beta}$ that is the closest to $\hat{v}$ ($\underline{\beta}$ is selected in the unlikely event of equality).

ii- Only $\delta_i(\underline{\beta})$ is free in $[0,1]$: Set α to $\underline{\beta}$.

iii- Only $\delta_i(\overline{\beta})$ is free in $[0,1]$: Set α to $\overline{\beta}$.

iv- Both $\delta_i(\underline{\beta})$ and $\delta_i(\overline{\beta})$ are fixed: Select α according to the corresponding minimization of error criterion of Proposition 4.4 or 4.5 (the latter criterion is used when the branching variable is associated to $|\hat{w}_{kj} - \hat{x}_k\hat{x}_j|$ and not to $|\hat{v}_i - \hat{x}_i^2|$). Add to the linear relaxation the variable $\delta_i(\alpha)$ and linearizations of class $\mathcal{C}_{III}$.

The branching value is α. ∎

Branching is done by creating two sons of the current node: one in which the variable $\delta_i(\alpha)$ is fixed at 0, and the other in which it is fixed at 1. In the first son, the variable x_i is constrained to be less than or equal to α, and in the second son it is forced to be greater than or equal to α. Note that this branching rule is discrepancy based; other such rules are discussed in Sherali and Tuncbilek [29], [29], [32].

5.3 Cutting

The objective of the cutting step of the algorithm is to refine as much as possible (up to the feasibility parameter ϵ_r) the outer-approximation using only linear cuts. The idea consists in iterating the four following steps, as long as at least one new linearization is added to the relaxation. The point $(\hat{x}, \hat{v}, \hat{w})$ still refers to the optimal solution of the current linear relaxation.

Optimality test:
If the optimal value of the relaxation is greater than the incumbent (minus ϵ_z), then no cutting is necessary, the current node may be discarded. Otherwise, the algorithm goes on with the next steps.

Refinement of an under-estimation of a square (class $\mathcal{C}_I$):
For each $i \in N$ such that $\hat{v}_i < \hat{x}_i^2 - \epsilon_r$, add the linearization $\underline{V}_i(\alpha)$ where α is defined through Proposition 4.1.

Refinement of a paraboloid (class $\mathcal{C}_{II}$):
For each $(i,j) \in M$ and γ such that the conditions in Proposition 4.2 are satisfied, add the linearization $\underline{P}_{ij}^{\gamma}(\alpha_i, \alpha_j)$, where (α_i, α_j) is defined through Proposition 4.3.

Refinement of a product (classes $\underline{\mathcal{C}}_{IV}$ and $\overline{\mathcal{C}}_{IV}$):
For each $(i,j) \in M$ such that $\hat{w}_{ij} < \hat{x}_i\hat{x}_j - \epsilon_r$, add the linearizations $\underline{W}_{ij}^{I}(\underline{\beta}, \overline{\beta})$ and $\underline{W}_{ij}^{III}(\underline{\beta}, \overline{\beta})$ if they are not already present (using the current variables $\delta_i(\cdot)$ and $\delta_j(\cdot)$), where $\underline{\beta}$ and $\overline{\beta}$ are evaluation points such that the interval $[\underline{\beta}, \overline{\beta}] \supseteq [\frac{\hat{w}_{ij}}{\hat{x}_j}, \hat{x}_i]$ is the smallest possible.
For each $(i,j) \in M$ such that $\hat{w}_{ij} > \hat{x}_i\hat{x}_j + \epsilon_r$, add the linearizations $\overline{W}_{ij}^{II}(\underline{\beta}, \overline{\beta})$ and $\overline{W}_{ij}^{IV}(\underline{\beta}, \overline{\beta})$ if they are not already present (using the current variables $\delta_i(\cdot)$ and $\delta_j(\cdot)$), where $\underline{\beta}$ and $\overline{\beta}$ are evaluation points such that the interval $[\underline{\beta}, \overline{\beta}] \supseteq [\hat{x}_i, \frac{\hat{w}_{ij}}{\hat{x}_j}]$ is the smallest possible. ∎

Note that if the relaxation is infeasible, then convention ensures that its optimal value will be equal to $+\infty$, and so the optimality test will stop processing the node.

The cuts of classes $\mathcal{C}_I$ and $\mathcal{C}_{II}$ are independent of the variables $\delta_i(\cdot)$ since they are linear under-approximations of convex functions. These cuts are inexpensive to add to the linear relaxation. Only the cuts associated to the classes $\mathcal{C}_{III}$, $\underline{\mathcal{C}}_{IV}$ and $\overline{\mathcal{C}}_{IV}$ use the variables $\delta_i(\cdot)$ that are already fixed at either 0 or 1. Cuts from $\mathcal{C}_{III}$ are only added and activated at the branching step. Cuts from the two other classes are added to the outer-approximation only when they are needed, thus keeping the size of the relaxation from growing.

5.4 Description of the whole Algorithm

We are now able to construct the branch and cut algorithm. The method can be divided in two main steps. The *Pre-Processing* step is used to create the initial outer-approximation and to obtain valid bounds on each variable. The *Enumeration Tree* step recursively processes nodes of the tree in a best-first manner (preference with respect to the optimal objective value of the relaxation). At each node, one of two outcomes

is possible: either the node is discarded (when infeasible, solved or eliminated), or it is split into two new nodes. For clarity, only the main ideas of the algorithm are presented. Details of the steps appear in the previous sections.

ALGORITHM.

Pre-Processing.
The list $\mathcal{L}$ of nodes to be explored is initialized to contain only the root node.

Enumeration Tree.
While $\mathcal{L}$ is not empty, repeat the three sub-steps.
- Select and remove the best-first node from $\mathcal{L}$.
- Perform the cutting steps:
  Add linearizations from classes $\mathcal{C}_I$, $\mathcal{C}_{II}$, $\underline{\mathcal{C}}_{IV}$ and $\overline{\mathcal{C}}_{IV}$.
  If the optimal relaxed solution is ϵ_r-feasible, then update the incumbent.
  Otherwise, pursue at the branching step if the relaxation is feasible and
  its optimal objective value is less than the incumbent objective value (minus ϵ_z).
- Perform the branching step.
  Obtain the branching variable x_i and value α and dichotomous variable $\delta_i(\alpha)$
  (if possible, reuse the structure created at other nodes).
  If the structure is not reused, introduce cuts from class $\mathcal{C}_{III}$.
  Add to $\mathcal{L}$ nodes corresponding to both sons. ∎

This is indeed a branch and cut algorithm in the sense that the cuts from classes $\mathcal{C}_I$ and $\mathcal{C}_{II}$ introduced at any node of the tree are valid at all other nodes. The cuts derived from the other classes are valid everywhere in the subtree rooted at the node where they were generated. At all other nodes, these cuts are relaxed as long as the corresponding variable δ_i is free in $[0,1]$. They are valid at all times, but become non-redundant as soon as the branching structure is reused, i.e., when the variable δ_i is fixed at either 0 or 1. The next theorem shows finiteness and correctness of the algorithm.

Theorem 5.2 *The above algorithm finds in finite time an ϵ_r-ϵ_z-optimal solution of problem QQP.*

Proof: The Pre-Processing step stops as soon as two successive iterations do not improve any bound by a value of at least ϵ_r. Only a finite number of iterations are therefore possible. At each iteration, a finite number of linear programs are solved. It follows that the Pre-Processing phase is completed in finite time.

Consider a node generated by the Enumeration Tree phase where the over-approximation of the square function is refined. Let $\alpha \in]\ell_i, u_i[$ be the point where the

linearization was done.

We now show that the linearizations associated to both sons of the current node eliminate a non-negligible region of the relaxed domain. For the node where $x_i \leq \alpha$, the linearization of the variable x_i is within the required tolerance if $x_i \geq \alpha - \epsilon_r/u_i$. So, if this condition is satisfied, the maximal error will be

$$(\ell_i + \alpha)(\alpha - \frac{\epsilon_r}{u_i}) - \ell_i\alpha - (\alpha - \frac{\epsilon_r}{u_i})^2 = \frac{\alpha\epsilon_r}{u_i} - \frac{\ell_i\epsilon_r}{u_i} - \frac{\epsilon_r^2}{u_i^2} \leq \frac{\alpha\epsilon_r}{u_i} \leq \epsilon_r.$$

For the son where $x_i \geq \alpha$, the linearization of the variable x_i is within the required tolerance if $x_i \leq \alpha + \epsilon_r/u_i$. So, if this condition is satisfied, the maximal error will be

$$(\alpha + u_i)(\alpha + \frac{\epsilon_r}{u_i}) - \alpha u_i - (\alpha + \frac{\epsilon_r}{u_i})^2 = \epsilon_r - \frac{\epsilon_r^2}{u_i^2} - \frac{\alpha\epsilon_r}{u_i} \leq \epsilon_r.$$

For each son an interval of length $\epsilon_r/u_i > 0$ is linearized within the error tolerance, and so, if a solution lying in that domain is generated, then the linearization will not be refined anymore. Therefore, there can only be a finite number of cuts for each variable.

The same reasoning applies for the approximation of a product of two variables. The solution produced will thus be ϵ_r-feasible.

The propositions from Section 3 imply that the inequalities are valid and so, the optimal solution (within the ϵ_z optimality tolerance) is never eliminated. It follows that there exists a node of the enumeration tree where an ϵ_r-ϵ_z-optimal solution will be identified.

Since the Pre-Processing and Enumeration Tree phases stop in finite time, the algorithm finds in finite time an ϵ_r-ϵ_z-optimal solution of QQP. ∎

In the following subsection, we illustrate the performance of this algorithm on different examples taken from the literature.

5.5 Numerical Results

The algorithm is coded in C++ and uses the CPLEX4.0 library to solve the linear programs. Computational experiments were made on a Ultra-1/167 station using Solaris 2.5-06.

The following example is a quadratic reformulation of a fourth degree polynomial problem found in Bartholomew-Biggs [8]. It is restated in Hock and Schittkowski [23]

(No 71) and in Hansen and Jaumard [19] (No 5). The reformulation (that introduces the variables x_5 and x_6) is stated as follows.

$$\begin{array}{ll} \min\limits_{x} & x_3 + x_1x_5 + x_2x_5 + x_3x_5 \\ \text{s.t.} & x_5 - x_1x_4 = 0, \\ & x_6 - x_2x_3 = 0, \\ & x_1^2 + x_2^2 + x_3^2 + x_4^2 = 40, \\ & x_5x_6 \geq 25, \\ & 1 \leq x_1 \leq 5 \qquad 1 \leq x_3 \leq 5, \\ & 1 \leq x_2 \leq 5 \qquad 1 \leq x_4 \leq 5. \end{array}$$

Using the precision $\epsilon_r = \epsilon_z = 10^{-6}$, the Pre-Processing phase spends 2.69 seconds to improve bounds on both variables x_5 and x_6 to $[1.33975, 18.6603]$. Then, the Enumeration Tree phase finds in 4.49 seconds the solution

$$x^* = (1, 4.74319, 3.8209, 1.37944, 1.37944, 18.1233)$$

having objective value 17.014 by exploring a total of 69 nodes, adding 25 variables $\delta_i(\alpha)$, and 599 linear cuts. The total time required to solve this problem is 7.18 seconds. In order to compare the usefulness of selecting the branching values using the error minimization criteria developed in Section 4, the same instance was solved by selecting the branching value to be the mid-point of the current interval. This strategy generated 189 nodes, added 55 variables $\delta_i(\alpha)$, and 1076 linear cuts.

Figure 3 illustrates aspects of the resolution. The enumeration tree is detailed only to a depth of six. The circled numbers indicate the order in which the nodes were generated. One can deduce from them the order in which they are processed (i.e., removed from the list $\mathcal{L}$). Directly below each node is the corresponding branching variable, except at node 12 where the relaxation was proven infeasible. The four numbers around each node represent the number of cuts of each class (as illustrated to the right of the tree) generated at the corresponding node. The dots under nodes 13 and 30 indicate that there are further branches in the tree underneath them. The other leaves (nodes 3, 7, 10, 20, 21, 18, 31) are not developed as their lower bound, that is, the optimal value of the relaxation, is greater than the incumbent value.

The enumeration tree suggests the following observations. The nodes are processed using a best-first strategy, and so the order in which they are selected is not predictable only by simple inspection of the tree.

Most of the cuts introduced come from the classes $\mathcal{C}_I$ and $\mathcal{C}_{II}$. That situation is desirable since these cuts correspond to linear under-estimations of convex functions and do not require dichotomy, hence no extra variable is needed and a single constraint

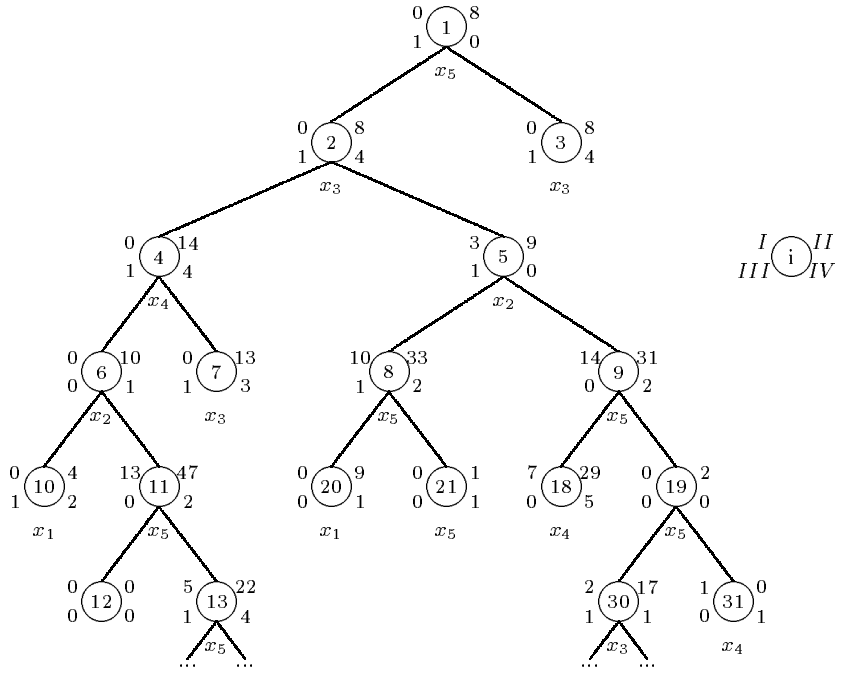

Figure 3: Partial branch and cut tree

is added to the outer-approximation. Moreover, the large number of cuts from class $\mathcal{C}_{II}$ is explained by the flexibility involved in their generation. The three required parameters allow frequent elimination of the current point.

Only some of the cuts available from classes $\underline{\mathcal{C}}_{IV}$ and $\overline{\mathcal{C}}_{IV}$ are added. This limits the growth in size of the relaxation.

At any node, there is never more than one new cut from class $\mathcal{C}_{III}$. This is expected since the branching process is invoked as soon as one such cut is added. When no cut of that class is introduced (nodes 6, 11, 20, 21, 18, 19, 31), a previously created branching variable and structure is reused. This is a key element to keep reasonable the size of the relaxation.

The algorithm also solved several global optimization problems from the literature. These problems were first reformulated as instances of QQP using techniques described in Hansen and Jaumard [19]. In particular, polynomial terms of degree more than two were rewritten, as in the example above, as quadratic ones. The same type of transformations was applied to fractions involving quadratic terms. Moreover, monotonicity analysis (as described in Hansen, Jaumard and Lu [20]) and variable elimination allowed significant simplifications of these problems. No comparison between

the several possible reformulations and their effect on the tightness of the relaxation was performed here. The final reformulations on which the algorithm was executed are detailed in Audet [5]. We present here the largest of these reformulations, i.e., problem 7 below, along with our ϵ_r-ϵ_z-optimal solution.

$$
\begin{array}{lll}
\min\limits_{x} & z(x) \;=\; 12.62626(x_{12}+x_{13}+x_{14}+x_{15}+x_{16}) & \\
 & \quad -1.231059(x_1x_{12}+x_2x_{13}+x_3x_{14}+x_4x_{15}+x_5x_{16}) & \\
\text{s.t.} & 50 \le z(x) \le 250, & \\
 & -3.475x_i + 100x_j + .0975x_i^2 - 9.75x_ix_j \le 0 & i=1,2,3,4,5, j=i+5, \\
 & -x_6x_{11} + x_7x_{11} - x_1x_{12} + x_6x_{12} \ge 0, & \\
 & 50x_7 - 50x_8 - x_1x_{12} + x_2x_{13} + x_7x_{12} - x_8x_{13} = 0, & \\
 & 50x_8 + 50x_9 - x_2x_{13} + x_3x_{14} + x_8x_{13} - x_9x_{14} \le 500, & \\
 & -50x_9 + 50x_{10} - x_3x_{14} + x_4x_{15} - x_8x_{15} + x_9x_{14} \le 0, & \\
 & 50x_4 - 50x_{10} - x_4x_{15} - x_4x_{16} + x_5x_{16} + x_{10}x_{15} \le 0, & \\
 & 50x_4 - x_4x_{16} + x_5x_{16} \ge 450 & -x_1 + 2x_7 \le 1, \\
 & x_1 \le x_2 \le x_3 \le x_4 \le x_5 & x_6 \le x_7 \qquad x_8 \le x_9 \le x_{10} \le x_4 \\
 & 0 \le x_{11} - x_{12} \le 50 & .001 \le x_6 \le 1, \\
 & 1 \le x_1 \le 8.03773157 & 1 \le x_7 \le 4.51886579 \qquad 10^{-7} \le x_{12} \le 100, \\
 & 1 \le x_2 \le 9 & 1 \le x_8 \le 9 \qquad 1 \le x_{13} \le 50, \\
 & 4.5 \le x_3 \le 9 & 1 \le x_9 \le 9 \qquad 50 \le x_{14} \le 100, \\
 & 4.5 \le x_4 \le 9 & 1 \le x_{10} \le 9 \qquad 50 \le x_{15} \le 100, \\
 & 9 \le x_5 \le 10 & .1 \le x_{11} \le 100 \qquad 10^{-7} \le x_{16} \le 50.
\end{array}
$$

Solution: $\epsilon_r = \epsilon_z = 10^{-5}$,

$$
x^* = (8.03773, 8.161, 9, 9, 9, 1, 1.07026, 1.90837, 1.90837,
1.90837, 50.5042, .504236, 7.26387, 50, 50, 0),\ z(x^*) = 174.788.
$$

Table 1 presents some important characteristics of these instances. The first column indicates the reference in which the first untransformed formulation of the problem appears, the other columns quantify aspects of the reformulated quadratic instances. The next three respectively contain the number of variables that are not involved in quadratic terms, the number of variables that are, and the total number of variables. The middle column indicates the total number of different quadratic terms. The last three presents the number of linear inequalities (including bounds on variables), and the number of nonlinear inequalities and equalities. There is no linear equality since the process of reformulating the problems into quadratic instances removed them by variable elimination.

The first example stems from the bilinear pooling problem encountered in the petrochemical industry. The next one describes a situation at *Proctor and Gamble Co.* The third one is a simple 3-stage heat exchanger design problem. The fourth

Ex	Source	Variables Lin	Quad	Tot	Quad Terms	Constraints Lin $\leq$	Quad $\leq$	$=$
1	Haverly [22]	2	3	5	2	7	2	0
2	Colville [11]	0	4	4	6	8	3	0
3	Avriel and Williams [6]	0	4	4	2	9	2	0
4	Bartholomew-Biggs [8]	0	6	6	9	8	1	3
5	Bracken and McCormick [9]	0	7	7	6	14	4	2
6	Dembo [12]	0	10	10	15	26	9	2
7	Dembo [12]	0	16	16	24	42	10	1

Table 1: Characteristics of the problems

one is more academic, and is the one described in details at the beginning of this section. The fifth one models an alkylation of olefin process. The two last ones arise from membrane separation in respectively three and five phases; the largest is written above. The diversity of these applications indicates the modeling flexibility of QQP.

Table 2 displays the result of the application of the algorithm to these problems. The column *bound* indicates the number of variables that are not, at optimality, at one of their bounds obtained in the pre-processing phase. Computational difficulty of a problem is closely related to that number since linearization is exact when the variable is at one of its bound. The columns entitled δ and *Add ctr* respectively contain the number of variables and constraints that were introduced in the solution process. The column *Nodes* shows the number of nodes required by the exploration tree. The columns *Time* indicate the pre-processing, the enumeration tree and the total time in seconds used by the algorithm. Finally, the last column displays the precision factors supplied to the algorithm.

Ex	Variables bound	δ	Add ctr	Nodes	Time (sec) PP	Tree	Tot	$\epsilon_r = \epsilon_z$
1	2	5	34	9	.64	.01	.65	10^{-6}
2	2	6	69	7	1.25	.04	1.29	10^{-6}
3	4	39	378	191	1.1	3.8	4.9	10^{-6}
4	5	25	599	69	2.7	4.5	7.2	10^{-6}
5	6	66	1335	357	4.3	65.6	69.9	10^{-5}
6	7	41	1088	259	148	61	209	10^{-5}
7	7	235	6205	2847	222	7329	7551	10^{-5}

Table 2: Performance of the algorithm

Even if the problems considered above arise from different sources and rely on different modeling structure, all of them are now solved to global optimality within a good precision. Prior to our work, other authors studied these problems. Our

algorithm confirmed that the heuristic solutions of problems 3 and 4 presented in Hock and Schittkowski [23] are indeed ϵ_r-ϵ_z-optimal, and it slightly improved that of problem 6. Problem 1 was solved by Foulds, Haugland and Jörnsten [17], but the tolerance parameter is not specified (however, their discussion implies that it is small). Problems 2 and 3 were solved to optimality by Hansen *et al.* [20] using monotonicity analysis together with a branch and bound scheme. Problem 5 was solved by Quesada and Grossmann [25] within a 5% tolerance using a branch and bound algorithm for fractional programs that uses convex nonlinear under-estimators. To the best of our knowledge, problems 6 and 7 are solved to ϵ_r-ϵ_z-optimality for the first time.

Based on the theoretical and algorithmic framework described herein, we intend in further work to pursue the study of such linearization approaches for other classes of problems, and to compare the results with alternate methods.

References

[1] AL-KHAYYAL F.A.(1990), "Jointly Constrained Bilinear Programs and Related Problems: An Overview," *Computers & Mathematics with Applications* 19, 53–62.

[2] AL-KHAYYAL F.A.(1992), "Generalized Bilinear Programming, Part I: Models, Applications and Linear Programming Relaxation," *European Journal of Operational Research* 60, 306–314.

[3] AL-KHAYYAL F.A. and FALK J.E.(1983), "Jointly Constrained Biconvex Programming," *Mathematics of Operations Research* 8, 273–286.

[4] AL-KHAYYAL F.A., LARSEN C. and VAN VOORHIS T.(1995), "A Relaxation Method for Nonconvex Quadratically Constrained Quadratic Programs," *Journal of Global Optimization* 6, 215–230.

[5] AUDET C.(1997), "Optimisation globale structurée: propriétés, équivalences et résolution," Thèse de Doctorat, École Polytechnique de Montréal.

[6] AVRIEL M. and WILLIAMS A.C.(1971), "An Extension of Geometric Programming with Applications in Engineering Optimization," *Journal of Engineering Mathematics* 5, 187–194.

[7] BENDERS J.F.(1962), "Partitioning Procedures for Solving Mixed-Variables Programming Problems," *Numerische Mathematik* 4, 238–252.

[8] BARTHOLOMEW-BIGGS M.C.(1976), "A Numerical Comparison Between Two Approaches to Nonlinear Programming Problems," *Technical Report #77*, Numerical Optimization Center, Hatfield England.

[9] BRACKEN J. and McCORMICK G.P.(1968), *Selected Applications of Nonlinear Programming* John Wiley and Sons, New York.

[10] CHUNG S.J.(1989), "NP-Completeness of the Linear Complementarity Problem," *Journal of Optimization Theory and Applications* 60, 393–399.

[11] COLVILLE A.R.(1970), *A Comparative Study of Nonlinear Programming Codes*, In *Princeton Symposium on Mathematical Programming*, KUHN H.W. (ed), Princeton University Press.

[12] DEMBO R.S.(1976), "A Set of Geometric Programming Test Problems and their Solutions," *Mathematical Programming* 10, 192–213.

[13] FLIPPO O.E.(1989), "Stability, Duality and Decomposition in General Mathematical Programming," Ph.D Thesis, Rotterdam: Erasmus University.

[14] FLIPPO O.E. and RINNOOY KAN A.H.G.(1993), "Decomposition in General Mathematical Programming," *Mathematical Programming* 60, 361–382.

[15] FLOUDAS C.A. and VISWESWARAN V.(1990), "A Global Optimization Algorithm (GOP) for Certain Classes of Nonconvex NLPs-I. Theory," *Computers and Chemical Engineering* 14, 1397–1417.

[16] FLOUDAS C.A. and VISWESWARAN V.(1993), "Primal-Relaxed Dual Global Optimization Approach," *Journal of Optimization Theory and Applications* 78, 237–260.

[17] FOULDS L.R, HAUGLAND D. and JÖRNSTEN K.(1992), "A Bilinear Approach to the Pooling Problem," *Optimization* 24, 165–180.

[18] GEOFFRION A.M.(1972), "Generalized Benders Decomposition," *Journal of Optimization Theory and Applications* 10, 237–260.

[19] HANSEN P. and JAUMARD B.(1992), "Reduction of Indefinite Quadratic Programs to Bilinear Programs," *Journal of Global optimization* 2, 41–60.

[20] HANSEN P., JAUMARD B. and LU S.(1991), "An analytical Approach to Global Optimization," *Mathematical Programming* 52, 227–254.

[21] HANSEN P., JAUMARD B. and SAVARD G.(1992), "New Branch-and-Bound Rules for Linear Bilevel Programming," *SIAM Journal on Scientific and Statistical Computing* 13, 1194–1217.

[22] HAVERLY C.A.(1978), "Studies of the Behaviour of Recursion for the Pooling Problem," *ACM SIGMAP Bulletin* 25, 19–28.

[23] HOCK W. and SCHITTKOWSKI K.(1981), *Test Examples for Nonlinear Programming Codes, Lecture Notes in Economics and Mathematical Systems ,187*, Springer-Verlag, Berlin, New-York.

[24] PHONG T.Q., TAO P.D. and HOAI AN L.T.(1995), "A Method for Solving D.C. Programming Problems. Application to Fuel Mixture Nonconvex Optimization Problem," *Journal of Global Optimization* 6, 87–105.

[25] QUESADA I. and GROSSMANN I.E.(1995), "A Global Optimization Algorithm for Linear Fractional and Bilinear Programs," *Journal of Global Optimization* 6, 39–76.

[26] RYOO H.S. and SAHINIDIS N.V.(1995), "Global Optimization of Nonconvex NLPs and MINLPs with Applications in process design," *Computers & Chemical Engineering* 19, 551–566.

[27] SHERALI H.D. and ALAMEDDINE A.(1990), "An Explicit Characterization of the Convex Envelope of a Bivariate Bilinear Function over Special Polytopes," *Annals of Operations Research* 25, 197–210.

[28] SHERALI H.D. and ALAMEDDINE A.(1992), "A New Reformulation-Linearization Technique for Bilinear Programming Problems," *Journal of Global Optimization* 2, 379–410.

[29] SHERALI H.D. and TUNCBILEK C.H.(1992), "A Global Optimization Algorithm for Polynomial Programming Using a Reformulation-Linearization Technique," *Journal of Global Optimization* 2, 101–112.

[30] SHERALI H.D. and TUNCBILEK C.H.(1995), "A Reformulation-Convexification Approach for Solving Nonconvex Quadratic Programming Problems," *Journal of Global Optimization* 7, 1–31.

[31] SHERALI H.D. and TUNCBILEK C.H.(1997), "Comparison of Two Reformulation-Linearization Technique Based Linear Programming Relaxations for Polynomial Programming Problems," *Journal of Global Optimization* 10, 381–390.

[32] SHERALI H.D. and TUNCBILEK C.H.(1997), "New Reformulation Linearization/Convexification Relaxations for Univariate and Multivariate Polynomial Programming Problems," *Operations Research Letters* 21, 1–9.

[33] SIMOES L.M.C.(1987), "Search for the Global Optimum of Least Volume Trusses," *Engineering Optimization* 11, 49–63.

[34] VISWESWARAN V. and FLOUDAS C.A.(1990), "A Global Optimization Algorithm (GOP) for Certain Classes of Nonconvex NLPs-II. Applications of Theory and Test Problems," *Computers and Chemical Engineering* 14, 1419–1434.

[35] WOLSEY L.A.(1981), "A Resource Decomposition Algorithm for General Mathematical Programs," *Mathematical Programming Study* 14, 244–257.